**Amendments to the Description:**

The description of the specification has been amended as follows to copy the textual description of sections IV through VIII from page 7 to page 24 with reference to Figures 3-13 in the U.S. Provisional Application No. 60/228,772 from which this application claims priority and the entirety of which incorporated by reference as part of this application.

A substitute specification is concurrently filed with this response to incorporate the following amendments.

1. Please insert the following paragraphs at the end of page 2 of the specification under the subtitle "Brief Description of the Drawings":

FIG. 1 is a schematic block diagram of a broadband system that includes a server and clients coupled by a network and can be used to implement the error correction techniques described in this application.

FIG. 2A illustrates an implementation of the passive recovery mode of the hybrid ARQ/FEC algorithm.

FIG. 2B illustrates an implementation of the active recovery mode of the hybrid ARQ/FEC algorithm.

FIG. 3 shows a worst-case retransmission scenario for the ARQ algorithm.

FIG. 4 illustrates a relation between the virtual playback and the original playback schedule.

FIG. 5 illustrates a worst-case retransmission scenario for the passive recovery mode of the hybrid ARQ/FEC algorithm.

FIG. 6 illustrates a worst-case retransmission scenario for the active recovery mode of the hybrid ARQ/FEC algorithm.

FIG. 7 shows traffic overhead at the server link versus packet-loss probability.

FIG. 8 shows the traffic overhead versus the number of receivers in the multicast session.

FIG. 9 shows the traffic overhead versus the loss limit tolerable by the media.

FIG. 10 shows the traffic overhead for parity group sizes ranging from 2 to 50.

FIG. 11 shows receiver buffer requirement versus parity group size.

FIG. 12 shows receiver buffer requirement versus packet-loss probability.

FIG. 13 shows traffic overhead versus heterogeneous packet-loss probabilities.

2. Please insert the following paragraphs beginning at page 11, line 1:

The following sections describe additional technical features of the error control in video distribution that combines both the ARQ and FEC error correction techniques.

Existing studies on Internet video delivery usually use overall traffic overhead in the entire multicast tree as a metric for performance evaluation. This is motivated by the fact that the Internet is known to have limited bandwidth and hence protocols that can minimize overall traffic overhead are highly desirable. By contrast, overall traffic overhead is less important in broadband residential network environments as the local or regional residential networks usually have relatively large bandwidth. The bottleneck is likely to be the trunk that connects the regional cluster with the central video server. This motivates us to consider traffic overhead at the server-

side network connection instead of over the entire multicast tree.

We define traffic overhead as the ratio between network traffic at the server link incurred in error recovery and the network traffic for normal video delivery. A traffic overhead of 0.1 means 10% additional network traffic incurred for the recovery of lost packets. Since the server link has only finite network bandwidth, too many traffic overheads will limit the scalability of the system. For simplicity, we ignore traffic overhead incurred by control packets such as NACK in the uplink from receivers to servers. This assumption is justified by the fact that retransmission requests are much smaller than retransmitted video packets. Hence, the downlink traffic volume will dominate the uplink traffic.

Unlike Internet video distribution where the receivers are often conventional computers with lots of memory, set-top boxes with limited memory are usually used in residential applications. Therefore we use receiver buffer requirement as the second metric for performance evaluation. Since video data is a continuous-media that must be presented at a predetermined rate, memory buffers are needed at the receiver to absorb delay and delay jitters incurred in video packet delivery. If no data is available for playback due to network delay or packet loss, video playback will be stalled - also known as playback starvation. On the other hand, if too many packets arrive at a receiver too early, receiver buffer overflow will occur and likely results in packet dropping. Since a receiver must prefill some of its buffers before playback starts, a startup delay is incurred whenever a receiver joins a video multicast

session. As this startup delay directly affects system responsiveness, the receiver buffer requirement should not be too large even if memory is abundant.

Let there be N receivers in a multicast session and let $p_i$ $(0 \le i \le N)$ be the probability of packet loss experienced by receiver i. To model network delay, we let $T_i$ be the average network delay, and $T_i^+$ $(T_i^+ \ge 0)$, $T_i^-$ $(T_i^- \le 0)$ be the maximum delay jitters between receiver i and the server. Hence, the network delay experienced by a packet must be within $(T_i+T_i^+)$ and $(T_i+T_i^-)$. The delay and delay jitters can be obtained a priori if the network have quality-of-service support.

At the receiver, let $p_{max}$ be the maximum tolerable packet-loss probability for the media (video). We will use the term loss limit to refer to $p_{max}$ in the rest of the paper. At the server, video packets of size $Q_s$ bytes are periodically multicasted with a period of $T_s$ seconds. Hence, if the video bit-rate is $R_v$ bytes-per-second (Bps), then $T_s=Q_s/R_v$. Let $T_{tran}$ be the transmission time for a video packet, i.e. the time for sending the entire video packet from the server memory onto the server-side network, with $T_{tran}<T_S$. Let $T_{start}$ be the time the server multicasts the first video packet for the multicast session. Then incorporating network delay, delay jitters, and transmission time, the arrival time at receiver i packet j $(j=0,1,2\ldots)$ denoted by $A_i^j$ is bounded by

$$\max\left\{T_{start}+T_{tran}+jT_S+T_i+T_i^-, T_{start}\right\} \le A_i^j \le T_{start}+T_{tran}+jT_S+T_i+T_i^+ \qquad (1)$$

Note that we need the maximization because Ti≤0. Knowing the above bounds, we can derive the performance metrics for ARQ, FEC, and the proposed hybrid algorithm in the following sections.

ANALYSIS OF AUTOMATIC REPEAT REQUEST (ARQ)

A. Traffic Overhead

Assume that packet losses are independent and with loss probability $p_i$, then the residual loss probability - the probability of an unrecoverable packet loss, after $K_i$ transmission attempts is simply given by $p_i^{K_i}$. To maintain video playback quality, we need to choose $K_i$ such that the loss limit is not exceeded:

$$p_i^{K_i} \le p_{\max} \tag{2}$$

Rearranging, we can then obtain $K_i$ from

$$K_i \ge \left\lceil \frac{\ln p_{\max}}{\ln p_i} \right\rceil \tag{3}$$

Given $K_i$, the expected number of transmissions for each video packet can be obtained from

$$\sum_{j=1}^{K_i} j\left(p_i^{j-1}(1-p_i)\right) \tag{4}$$

Hence, the ratio of extra traffic overhead incurred (excluding the first transmission) in retransmission for receiver i is given by

$$h_i = \frac{\text{traffic for retransmission}}{\text{traffic for data}} = \frac{\sum_{j=1}^{K_i}(j-1)\left(p_i^{j-1}(1-p_i)\right)}{1} \quad (5)$$

Finally, the traffic overhead at the server link is just the sum of traffic overhead for all receivers:

$$H_{ARQ} = \sum_{i=0}^{N-1} h_i \quad (6)$$

B. Receiver Buffer Requirement

To determine the receiver buffer requirement, we let $L_{ARQ}(i)=Y_i+Z_i$ be the number of buffers (each $Q_s$ bytes) in receiver i, managed as a circular buffer. The receiver starts playback once $Y_i$ buffers are filled with video data. These $Y_i$ prefetched buffers are then used to absorb delay variations in packet arrivals to prevent video playback starvation (i.e. buffer underflow). On the other hand, we reserve $Z_i$ empty buffers to cater for early-arrival packets to prevent buffer overflow.

Under this model, video playback effectively starts at time $A_i^{y_i-1}$, which is the time video packet ($Y_i$-1) arrives at receiver i. Hence, the playback time for video packet j of receiver i is

$$P_i^j = A_i^{Y_i-1} + jT_S \tag{7}$$

For simplicity, we assumed a constant playback time ($T_s$) for all video packets. This simplification can be removed to incorporate variable (but bounded) playback time or even variable-bit-rate video playback using techniques in [12].

To maintain video playback continuity, we must ensure that all video packets arrive before playback deadline. Formally, the arrival time for a video packet must be earlier than this playback time:

$$A_i^j \le P_i^j \quad \forall j \tag{8}$$

Substituting the upper bound for $A_i^j$ and the lower bound for $P_i^j$ into (8), we can then obtain the condition for continuity as

$$\begin{aligned} \max\{A_i^j\} &\le \min\{P_i^j\} \\ T_{start} + T_{tran} + jT_S + T_i + T_i^+ &\le \min\{A_i^{Y_i-1} + jT_S\} \\ &= \min\{A_i^{Y_i-1}\} + jT_S \\ &= T_{start} + T_{tran} + (Y_i - 1)T_S + T_i + T_i^- + jT_S \end{aligned} \tag{9}$$

Rearranging we can obtain $Y_i$ from

$$Y_i \ge \frac{T_i^+ - T_i^-}{T_S} + 1 \tag{10}$$

Similarly, to prevent buffer overflow, we need to ensure that an empty buffer is always available when a new video packet arrives. As the receiver buffers are managed as a circular buffer, we need to ensure that

$$\begin{aligned} &\min\left\{A_i^{j+Y_i+Z_i-1}\right\} \geq \max\left\{P_i^j + T_S\right\} \\ &T_{start} + T_{tran} + (j + Y_i + Z_i - 1)T_S + T_i + T_i^- \geq T_{start} + T_{tran} + (j + Y_i)T_S + T_i + T_i^+ \end{aligned} \tag{11}$$

Rearranging, we can then obtain $Z_i$:

$$Z_i \geq \frac{T_i^+ - T_i^-}{T_S} + 1 \tag{12}$$

Hence, the total number of receiver buffers needed for receiver i is:

$$L_{ARQ}(i) = 2\left\lceil \frac{\left(T_i^+ - T_i^-\right)}{T_S} \right\rceil + 2 \tag{13}$$

To incorporate the effect of retransmission, we note that each retransmission attempt incurs a maximum additional delay of $T_{tran}+T_{wait}+T_i+T_i^+$, where $T_{wait}$ is the retransmission timeout (Fig. 3). Since at most $K_i$ transmissions (including retransmissions) are attempted for delivering a video packet, the upper bound for the arrival time of packet j of receiver i is modified to

$$A_i^j \leq \left(T_{start} + T_{tran} + jT_S + T_i + T_i^+\right) + \left(K_i - 1\right)\left(T_{tran} + T_{wait} + T_i + T_i^+\right) \tag{14}$$

where the first term is the worst-case delay incurred in the first transmission and the second term is the worst-case delay incurred in the next ($K_i$-1) retransmissions.

Again, let there be $L_{ARQ}(i)=Y_i+Z_i$ receiver buffers, and playback starts after $Y_i$ buffers are filled with video data. Now as the packet-loss probability is non-zero, it is possible that some of those first $Y_i$ packets are lost and requires retransmissions. In particular, if video packet ($Y_i$=1) is lost, the prefetch process (and hence playback) will be delayed. To avoid unnecessary delay, the receiver starts playback when the expected arrival time for packet (prefetch process (and hence playback) will be delayed. To avoid unnecessary delay, the receiver starts playback when the expected arrival time for packet ($Y_i$-1)

$$P_i^0 = A_i^0 + (Y_i - 1)T_S \tag{15}$$

is reached, regardless of whether the packet has physically arrived or not (i.e. due to loss or delay jitter). Hence in general, we have

$$P_i^j = A_i^0 + (Y_i + j - 1)T_S \tag{16}$$

Using the continuity condition, we can obtain $Y_i$ from

$$\begin{aligned} \max\{A_i^j\} &\le \min\{P_i^j\} \\ T_{start}+T_{tran}+jT_S+T_i+T_i^+ +(K_i-1)(T_{tran}+T_{wait}+T_i+T_i^+) &\le \min\{A_i^0+(Y_i+j-1)T_S\} \\ &= T_{start}+T_{tran}+T_i+T_i^- +(Y_i+j-1)T_S \end{aligned} \tag{17}$$

Rearranging, we then have:

$$Y_i \ge \frac{(K_i-1)(T_{tran}+T_{wait}+T_i+T_i^+)+(T_i^+-T_i^-)}{T_S}+1 \tag{18}$$

Compared to the case without packet loss (c.f. Eq. (10)), more buffers are needed to absorb the extra delay incurred in retransmissions.

Similarly, to prevent buffer overflow, we need to ensure that:

$$\begin{aligned} \min\{A_i^{j+Y_i+Z_i-1}\} &\ge \max\{P_i^j+T_S\} \\ T_{start}+T_{tran}+(j+Y_i+Z_i-1)T_S+T_i+T_i^- &\ge \max\{A_i^0+(Y_i+j)T_S+T_i\} \\ &= T_{start}+T_{tran}+T_i+T_i^+ +(Y_i+j)T_S \end{aligned} \tag{19}$$

Rearranging, we can then obtain $Z_i$ as well:

$$Z_i \ge \frac{(T_i^+-T_i^-)}{T_S}+1 \tag{20}$$

Note that $Z_i$ is the same as the case without packet loss in (12). This is because retransmissions make packets arrive only

later but never earlier. Summing (18) and (20) we can then obtain the receiver buffer requirement for receiver i:

$$L_{ARQ}(i)=\left\lceil \frac{(K_i-1)\left(T_{tran}+T_{wait}+T_i+T_i^{+}\right)+\left(T_i^{+}-T_i^{-}\right)}{T_S} \right\rceil+\left\lceil \frac{T_i^{+}-T_i^{-}}{T_S} \right\rceil+2 \qquad (21)$$

If the network delay and delay jitters are known a priori, then the retransmission timeout $T_{wait}$ can simply be set to equal to the maximum network delay ($T_i + T_i^{+}$) and (21) can be simplified to

$$L_{ARQ}(i)=\left\lceil \frac{(K_i-1)\left(T_{tran}+2\left(T_i+T_i^{+}\right)\right)+\left(T_i^{+}-T_i^{-}\right)}{T_S} \right\rceil+\left\lceil \frac{T_i^{+}-T_i^{-}}{T_S} \right\rceil+2 \qquad (22)$$

As an example, using the parametric values in Table 1, the buffer requirement as calculated from (22) will be 41 units. As each unit stores a packet of size 1024 bytes, total buffer requirement would be 41KB.

Note that in practice, one would implement the receiver such that NACK is sent as soon as a gap in sequence number is detected. However, the time to detetct such a gap depends on a number of parameters, namely packet size, video bit-rate, network delay jitter, and the extent of burst losses. Hence without resorting to specific parameter ranges, it would be difficult (if possible) to derive a general formula for the delay incurred in detecting a sequence-number gap. For special cases where gap-based detection delay is known to be shorter than the maximum network delay, one can simply replace $T_{wait}$ with

the appropriate formula for more accurate results. For simplicity, we will assume $T_{wait} = (T_i+T_i^+)$ in the rest of the paper.

C. Server Buffer Requirement

At the server, buffers must be allocated to store outgoing video packets temporarily to support retransmissions. To determine the amount of additional buffer needed to support retransmission, we notice that the $n^{th}$ (n=1,2,...$K_i$-1) NACK for a transmitted video packet will arrive at the server at most

$$r_i^n \le n\left(T_{tran} + T_{wait} + T_i + T_i^+\right) \tag{23}$$

seconds after the first transmission attempt (c.f. Fig. 3). Hence, the maximum buffer-occupation time for a video packet is simply given by $r_i^{K_i-1}$. Assuming that video packets for transmission are generated at the same rate as consumption, i.e. one packet per $T_s$ seconds, the number of video packets generated during this duration $r_i^{K_i-1}$ is then given by

$$b_i = \left\lceil \frac{r_i^{K_i-1}}{T_S} \right\rceil \tag{24}$$

To cater for the requirements of all clients, the amount of buffer required for a server to support retransmission for one

multicast video stream would then be the maximum among all clients:

$$B_{ARQ} = Q_S \max\{b_i \mid i = 0,1,\ldots,(N-1)\} \quad (25)$$

ANALYSIS OF FORWARD ERROR CORRECTION (FEC)

In FEC, redundant data are introduced into the data stream in the form of redundant packets. Specifically, for every D video packets, R redundant packets are appended for transmission to the receivers. These R redundant packets are computed from the D video pockets using erasure-correction codes such as parity codes, or Reed-Solomon codes to form a parity group [13]. As R extra redundant packets are transmitted for every parity group, the packet inter-departure time at the server must be shortened to maintain the original video bit-rate. The new inter-departure time $\tilde{T}_S$ can be obtained from

$$\tilde{T}_S = T_S \frac{D}{D+R} \quad (26)$$

Note that while the packet inter-departure time is shortened, the average rate at which video packets are transmitted remains the same. The extra transmissions are redundant packets computed from a parity group of video packets.

A. Traffic Overhead

With R redundant packets, a receiver can correctly recover all D all video packets as long as no more than R packets are lost within the same parity group. However, in the event that more than R packets are lost, a decoding error [13] will occur. In the worst case, all D video packets will be lost. Hence, the upper bound for the residual loss probability given by

$$\varepsilon_i = \sum_{k=R+1}^{D+R} \binom{D+R}{k} p_i^k (1-p_i)^{R+D-k} \tag{27}$$

On the other hand, if the erasure-correction code employed is systematic, i.e. the D video packets are not modified during the encoding process, then the residual packet-loss probability will be smaller. Let there by k lost packets among the D video packets and R redundant packets. Then on average

$$k\frac{D}{D+R} \tag{28}$$

Of the lost packets will be video packets. As there are D video packets, the resultant loss probability for video packets given there are k losses is simply equal to

$$k\frac{D}{D+R}\cdot\frac{1}{D} = \frac{k}{D+R} \tag{29}$$

Hence, we can obtain the residual loss probability by conditioning on k:

$$\varepsilon_i = \sum_{k=R+1}^{D+R} \binom{D+R}{k} p_i^k (1-p_i)^{R+D-k} \frac{k}{D+R} \tag{30}$$

To maintain a residual loss probability of no more than $p_{max}$, we need a redundancy of at least

$$R_{FEC} = \min\{R \mid \varepsilon_i \le p_{\max}, \forall i\} \tag{31}$$

Using a redundancy of $R_{FEC}$, the traffic overhead at the server link can be obtained from

$$H_{FEC} = \frac{R_{FEC}}{D} \tag{32}$$

Note that unlike ARQ, traffic overhead incurred by FEC is fixed irrespective of the number of receivers in the system.

B. Receiver Buffer Requirement

To determine the receiver buffer requirement, we again let $L_{FEC}(i)=Y_i+Z_i$ be the number of buffers (each $Q_s$ bytes) in receiver i and assume playback starts once $Y_i$ buffers are filled with video data. However, as some of the packets are redundant packets, these redundant packets are not "played back" in the usual sense. In particular, the redundant packets will be used for erasure correction if there are packet losses; or discarded otherwise. To account for this, we create a virtual playback schedule where the "playback" duration of a video packet or a

redundant packet is $\tilde{T}_s$ rather than $T_s$ (Fig. 4). Under this virtual playback schedule where redundant packets are being consumed by the receiver just like normal video packets, all (except the first video packet in a parity group) video packets' playback deadlines are pushed forward and continuity in this new schedule implies continuity in the original schedule.

Formally, the video playback starts at time $A_i^{Y_i-1}$, which is the time video packet ($Y_i$-1) arrives at receiver i. Hence the playback time for video packet j is

$$P_i^j = A_i^{Y_i-1} + jT_s \tag{33}$$

To maintain video playback continuity, we must ensure that all video packets from a parity group arrive before the playback deadline of the first packet in the same parity group is reached[3]. For example, let packet j be the next packet for playback and it is lost in transit. In the worst case, this packet will be the first packet of a parity group and we need the entire group to recover it. Since packet (j+R+D-1) would be the last packet in the parity group, we need to ensure that it arrives before the playback deadline for packet j:

$$A_i^{j+R+D-1} \le P_i^j \quad \forall j \tag{34}$$

Substituting the upper bound for $A_i^{j+R+D-1}$, and the lower bound for $P_i^j$ into (34) we can then obtain the condition for continuity as

$$\begin{aligned} \max\left\{A_i^{j+R+D-1}\right\} &\le \min\left\{P_i^j\right\} \\ T_{start}+T_{tran}+(j+R+D-1)\tilde{T}_S+T_i+T_i^+ &\le \min\left\{A_i^{Y_i-1}+j\tilde{T}_S\right\} \\ &= \min\left\{A_i^{Y_i-1}\right\}+j\tilde{T}_S \\ &= T_{start}+T_{tran}+(Y_i-1)\tilde{T}_S+T_i+T_i^-+j\tilde{T}_S \end{aligned} \tag{35}$$

Rearranging gives the requirement for $Y_i$ as

$$Y_i \ge \frac{\left(T_i^+ - T_i^-\right)}{\tilde{T}_S}+D+R \tag{36}$$

Similarly, we can obtain $Z_i$ using the same derivations as in Section V-B:

$$Z_i \ge \frac{\left(T_i^+ - T_i^-\right)}{\tilde{T}_S}+1 \tag{37}$$

The total receiver buffer requirement for receiver i is just the sum of $Y_i$ and $Z_i$:

$$L_{FEC}(i) = 2\left\lceil \frac{\left(T_i^+ - T_i^-\right)}{\tilde{T}_S} \right\rceil + D + R + 1 \tag{38}$$

C. Server Buffer Requirement

Unlike ARQ, additional server buffers are not required to support retransmissions. However, depending on the erasure-correction code employed and the way the code is computed, additional buffers may be required to support encoding.

We first consider the worst-case scenario where redundant packets are computed on-the-fly. Additionally, we assume that video packets of a parity group must all be available before encoding. Then for a system with D video packets and R redundant packets per parity group, the server will need to allocate (D+R) buffer units for encoding. Hence the amount of buffer required for one multicast video stream is

$$B_{FEC} = (D+R)Q_s \tag{39}$$

This additional buffer requirement can be reduced in two ways. The first way is to use special erasure-correction codes that process video packets sequentially. For example, the parity coding scheme for R=1 can be computed on a packet-by-packet basis. Only one packet's worth of buffer is required to store the intermediate computation results while buffers for transmitted video packets can be released immediately. This translates into a buffer requirement of only one packet per stream (i.e. $B_{FEC}=Q_s$).

The second way is to use pre-computed redundant packets. For example, the redundant packets can be computed beforehand and stored along video data in the disks. In this way, no encoding needs to be performed and hence no additional buffer is

required. The tradeoffs are overheads in disk storage and disk I/O bandwidth.

ANALYSIS OF HYBRID ARQ/FEC

A. Passive Recovery

In passive recovery, the probability of a lost packet being unrecoverable by FEC is given by (30), i.e. the residual loss probability $\varepsilon_i$. The amount of traffic overhead incurred by ARQ would then be given by (5) and (6), with $p_i$ replaced by the corresponding residual loss probability $\varepsilon_i$. Hence, given a redundancy of R, the traffic overhead can be computed from

$$H_{hybrid}(R) = \frac{R}{D} + \sum_{i=0}^{N-1}\left(\sum_{j=1}^{K_i}(j-1)\left(\varepsilon_i^{j-1}(1-\varepsilon_i)\right)\right) \qquad (40)$$

$$\text{where } K_i = \left\lceil \frac{\ln p_{\max}}{\ln \varepsilon_i} \right\rceil.$$

---

The level of redundancy R essentially determines how many losses are to be recovered through FEC, while the rest will be recovered by ARQ. The key is that traffic overhead incurred in FEC is fixed irrespective of number of receivers while overhead for ARQ increases with number of receivers. Hence, for systems with many receivers, the extra overhead in increasing R can be more than offset by the corresponding reduction in ARQ overhead. From (31), the maximum amount of redundancy needed is $R_{FEC}$. Hence the optimal amount of redundancy for hybrid ARQ/FEC, denoted by $R_{hybrid}$, required to minimize the traffic overhead as

given by (40) can be determined by choosing $R_{hybrid}=0,1,\ldots,R_{FEC}$ such that $H_{hybrid}(R_{hybrid})$ is minimized.

Note that this hybrid ARQ/FEC algorithm reduces to ARQ when $R_{hybrid}=0$ and reduces to FEC when $R_{hybrid}=R_{FEC}$. Furthermore, as we choose $R_{hybrid}$ such that $H_{hybrid}(R_{hybrid})$ is minimized, this hybrid algorithm performs at least as good as either ARQ or FEC alone.

At the receiver, we need to reserve buffer space not only for erasure correction, but also for retransmissions as well. Fig. 5 depicts the worst-case scenario for retransmitting packets not recoverable by erasure correction. Since retransmission starts after the entire parity group is received, the receiver can send a single request back to the server for all lost packets in the parity group. If one or more retransmitted packets are lost again, the next round of retransmission starts immediately after the last retransmitted packet has timed out in the current round. This process repeats until either all packets are received or the maximum number of retransmission attempts is reached.

At the server, we assume that retransmitted packets are sent at normal data rate, i.e. with a transmission time of $T_{tran}$, and an inter-packet time of (equation). The server $\tilde{T}_s$. The server deliberately smooths out the retransmission traffic to avoid causing congestion in the network or a t the receiver.

As the receiver will attempt at most $(K_i-1)$ retransmission for a video packet, the worst-case delay incurred for receiving any packet within the parity group is given by see (Fit. 5):

$$\left[(D+R-1)\tilde{T}_S+T_{tran}+T_i+T_i^+\right]+(K_i-1)\left[T_i+T_i^++(D-1)\tilde{T}_S+T_{tran}+T_i+T_i^+\right] \quad (41)$$

where the first term is the worst-case delay incurred in waiting for the whole parity group to arrive and the second term is the worst-case delay incurred in those ($K_i$-1) retransmission assuming all D video packets are lost in every transmission (and retransmission) rounds except the last one. Hence, the upper bound for $A_i^j$ becomes:

$$A_i^j \le \left(T_{start}+j\tilde{T}_S+(D+R-1)\tilde{T}_S+T_{tran}+T_i+T_i^+\right)+(K_i-1)\left((D-1)\tilde{T}_S+T_{tran}+2(T_i+T_i^+)\right) \quad (42)$$

Together with the continuity condition in (8) and (9), we can obtain $Y_i$ from:

$$\max\{A_i^j\} \le \min\{P_i^j\} \quad (43)$$

Substituting (42) for the L.H.S. and (16) for the R.H.S. we can then obtain the requirement on $Y_i$:

$$Y_i \ge \frac{(K_i-1)\left((D-1)\tilde{T}_S+T_{tran}+2(T_i+T_i^+)\right)+(T_i^+-T_i^-)}{\tilde{T}_S}+D+R \quad (44)$$

Again as the lower bound for $A_i^j$ is not changed, the requirement for $Z_i$ is the same as FEC:

$$Z_i \geq \frac{\left(T_i^+ - T_i^-\right)}{\tilde{T}_S} + 1 \qquad (45)$$

The receiver buffer requirement for receiver i is then given by

$$L_{hybrid}(i) = \left\lceil \frac{(K_i - 1)\left((D-1)\tilde{T}_S + T_{tran} + 2\left(T_i + T_i^+\right)\right) + \left(T_i^+ - T_i^-\right)}{\tilde{T}_S} \right\rceil + \left\lceil \frac{T_i^+ - T_i^-}{\tilde{T}_S} \right\rceil + D + R + 1 \qquad (46)$$

At the server, two types of additional buffers are needed to support passive recovery. First (D+R) buffers are needed to support erasure-correction encoding. Second, buffers holding transmitted packets cannot be released until the (i.e. $K_i$-1) retransmission request arrives. According to Fig. 5, the client starts requesting retransmission after the whole parity group is received, i.e. after a maximum delay of (c.f. Fig. 5).

$$\tilde{r}_i = (D + R - 1)\tilde{T}_S + T_{tran} + T_i + T_i^+ \qquad (47)$$

Hence, the delay for the first retransmission request to arrive at the server is at most

$$r_i^1 = \tilde{r}_i + \left(T_i + T_i^+\right) \qquad (48)$$

Similarly, delay for the $n^{th}$ retransmission request to arrive is at most

$$r_i^n = \tilde{r}_i + \left(T_i + T_i^+\right) + (n-1)\left(T_i + T_i^+ + (D-1)\tilde{T}_S + T_{tran} + T_i + T_i^+\right) \quad (49)$$

Given that new groups of D packets are being generated at a rate of one group per $DT_S$, there will be at most

$$\left\lceil \frac{r_i^{K_i-1}}{DT_S} \right\rceil \quad (50)$$

groups co-existing simultaneously for a multicast video stream. Together with redundant packets, each parity group requires $(D+R)Q_S$ bytes of buffer and hence the total buffer requirement can be obtained from

$$B_{hybrid} = Q_S(D+R)\left\lceil \frac{r_i^{K_i-1}}{DT_S} \right\rceil \quad (51)$$

Active Recovery

With active recovery, retransmission is needed only if there are more than R lost packets because otherwise the lost packets would be recoverable by erasure correction already. For example, if there are exactly m packets lost in a parity group, only (m-R) of the m lost packets need to be retransmitted. On the other hand, retransmission can start as soon as the receiver detects the loss of the $(R+1)^{th}$ packet without waiting for the whole parity group to arrive. We assume that given there are m (m>R) packets lost in the first transmission of a parity group,

the receiver will request retransmission for the lat (m-R) of these lost packets. As each one of these (m-R) packets will be retransmitted for at most ($K_i$-1) times, the probability that a retransmitted packet cannot be recovered is equal to $p_i^{K_i-1}$. Hence, the probability that w of the (m-R) packets being recoverable by retransmission is given by

$$u_i(w) = \binom{m-R}{w} \left(p_i^{K_i-1}\right)^w \left(1-p_i^{K_i-1}\right)^{m-R-w}, \quad (m-R) \geq w \geq 0, K_i > 1 \tag{52}$$

Using this equation, we can compute the conditional probability of having k(m≥k≥R) lost packets in the parity group after retransmissions given there are m lost packets from

$$\Phi_i(k,m) = u_i(k-R) = \binom{m-R}{k-R} \left(p_i^{K_i-1}\right)^{k-R} \left(1-p_i^{K_i-1}\right)^{m-k}, \quad m \geq k \geq R \tag{53}$$

Now, the probability that there are exactly m packets lost in a parity group is given by

$$\Pr\{m \text{ lost in a parity group}\} = \binom{D+R}{m} p_i^m (1-p_i)^{D+R-m} \tag{54}$$

Therefore we can obtain the unconditional probability of having k (k≥R) lost packets after retransmission from

$$\Phi_i(k) = \sum_{m=k}^{D+R} \Phi_i(k,m) \Pr\{m \text{ lost in a parity group}\}, \quad k \geq R$$
$$= \sum_{m=k}^{D+R} \binom{m-R}{k-R} \left(p_i^{K_i-1}\right)^{k-R} \left(1-p_i^{K_i-1}\right)^{m-k} \binom{D+R}{m} p_i^m (1-p_i)^{D+R-m}, \quad k \geq R \tag{55}$$

Now if k>R, then erasure correction will not succeed. Assuming the erasure-correction code is systematic, then the loss probability for video packets in a parity group is given by (29). Hence, we can obtain the residual loss probability by conditioning on k:

$$\varepsilon_i(R) = \sum_{k=R+1}^{D+R} \frac{k}{D+R} \Phi_i(k) \tag{56}$$

To determine the traffic overhead, we first need to find $K_I$. Given R and the loss limit $p_{max}$, we can compute the minimum value for $K_i$ such that the condition

$$\varepsilon_i(R) \leq p_{\max} \tag{57}$$

is satisfied. Once $K_i$ is known, we can obtain the amount of traffic overhead incurred in retransmission for each lost packet from

$$h_i = \sum_{j=1}^{K_i-1} j\left(p_i^{j-1}(1-p_i)\right) \tag{58}$$

Hence given there are m(m>R) lost packets in a parity group, the expected amount of traffic overhead incurred is given by $(m-R)h_i$. The average amount of traffic overhead incurred by all N receivers can then be obtained from

$$h(R)=\frac{1}{D}\sum_{i=0}^{N-1}\sum_{m=R+1}^{D+R}(m-R)h_i\binom{D+R}{m}p_i^m(1-p_i)^{D+R-m}$$
$$=\frac{1}{D}\sum_{i=0}^{N-1}\sum_{m=R+1}^{D+R}\sum_{j=1}^{K_i-1}\binom{D+R}{m}j(m-r)p_i^{m+j-1}(1-p_i)^{D+R-m+1} \quad (59)$$

and the total traffic including redundancy is

$$H_{hybrid}(R)=\frac{R}{D}+h(R) \quad (60)$$

To minimize the total server bandwidth requirement, one can choose $R_{hybrid}=0,1,\ldots,R_{FEC}$ such that $H_{hybrid}R_{hybrid}$ is minimized.

To determine the receiver buffer requirement for video playback continuity, we note that the receiver starts requesting retransmission as soon as (R+1) packet losses are detected. Hence, the worst-case arrival time (i.e. after $(K_i-1)$ retransmissions) for packet x must be larger than y if x>y. In the worst-case scenario, all packets (except those not retransmitted) must have arrived (or will never arrive) by the time the last packet of the parity group arrives after $(K_i-1)$ retransmissions (see Fig. 6). Therefore the packet arrive time is bounded from above by

$$A_i^j \le \left(T_{start} + j\tilde{T}_S + (D+R-1)\tilde{T}_S + T_{tran} + T_i + T_i^+\right) + \left(K_i - 1\right)\left(T_{tran} + 2(T_i + T_i^+)\right) \quad (61)$$

Using similar derivations, we can obtain $Y_i$ as

$$Y_i \ge \frac{(K_i - 1)\left(\tilde{T}_S + T_{tran} + 2\left(T_i + T_i^+\right)\right) + \left(T_i^+ - T_i^-\right)}{\tilde{T}_S} + D + R \quad (62)$$

And $Z_i$ as in (45). The total receiver requirement becomes

$$L_{hybrid}(i) = \left\lceil \frac{(K_i - 1)\left(\tilde{T}_S + T_{tran} + 2\left(T_i + T_i^+\right)\right) + \left(T_i^+ - T_i^-\right)}{\tilde{T}_S} \right\rceil + \left\lceil \frac{T_i^+ - T_i^-}{\tilde{T}_S} \right\rceil + D + R + 1 \quad (63)$$

To determine the amount of server buffer required to support active recovery, we notice that delay for the $n^{th}$ retransmission of the last packet in a parity group is given by

$$r_i^n = \tilde{r}_i + \left(T_i + T_i^+\right) + (n-1)\left(T_{tran} + 2(T_i + T_i^+)\right) \quad (64)$$

Using derivations similar to Section VII-A, the server buffer requirement can be found to be

$$B_{hybrid} = Q_S(D+R)\left\lceil \frac{r_i^{K_i - 1}}{DT_S} \right\rceil \quad (65)$$

PERFORMANCE EVALUATION

In this section, we compare the performance of the proposed hybrid ARQ/FEC algorithm with ARQ and FEC with respect to various system parameters. The values of the system parameters used for numerical evaluation are summarized in Table 1.

A. Traffic Overhead versus Packet-Loss Probability

Fig. 7 plots the traffic overhead at the server link versus packet-loss probabilities ranging from 0.001 to 0.1. The main observation here is that ARQ incurs far more traffic overhead than either FEC or hybrid ARQ/FEC for both 100 and 1000 receivers. This illustrates the well-known scalability problem in using ARQ for multicast data distribution. Secondly, we observe that both variants of the proposed hybrid algorithms performs equal to or better than the FEC algorithm, which in turns outperform ARQ by a large margin. Thirdly among the two hybrid variants, active recovery performs best. This result shows that the proposed hybrid algorithm is superior to both ARQ and FEC for a wide range of packet-loss probabilities.

B. Traffic Overhead versus Number of Receivers

Fig. 8 plots the traffic overhead versus the number of receivers in the multicast session. We observe that ARQ performs better than FEC when the number of receivers is smaller than 32, and FEC performs better otherwise. On the other hand, the proposed hybrid algorithms perform equal to or better than both ARQ and FEC, with active recovery performs slightly better. Hence, the proposed algorithm not only can be scaled up to serve a large number of receivers, but also can maintain a low traffic overhead when the number of receivers is small.

C. Traffic Overhead versus Loss Limit

Fig. 9 plots the traffic overhead versus the loss limit tolerable by the media. The result shows that FEC performs better than ARQ, and the hybrid algorithms perform even better for a wide range of loss limits. Once again, the active recovery scheme performs slightly better than the passive recovery scheme due to the better utilization of the redundant packets.

D. Traffic Overhead versus Parity Group Size

Fig. 10 plots the traffic overhead for parity group sizes ranging from 2 to 50. Obviously, ARQ is independent of the parity group size and hence remains constant in this figure. For FEC and the hybrid algorithms, increasing the parity group size generally reduces the amount of traffic overhead at the server. This is expected because the traffic overhead is directly proportional to the ration R/D as expressed in (32), (40) and (60). However, the parity group size also affects the receiver buffer requirement as well (Fig. 11). Interestingly, increasing the parity group size does not necessary raise the receiver buffer requirement. In some cases, like going from D=10 to D=12 in Hybrid-Passive with 100 receivers, the buffer requirement decreases substantially for the increase in parity group size. This observation is explained by the fact that the increase in parity group size results in a reduction in the need for retransmission. For the Hybrid-Passive with 100 receivers case, increasing D=10 to D=12 also increases the optimal amount of redundancy from R=1 to R=2. This further decreases the

maximum number of transmissions needed from $K_i$=2 to $K_i$=1 (i.e. FEC only), which reduces the receiver buffer requirement substantially.

E. Receiver Buffer Requirement versus Packet-Loss Probability

Fig. 12 plots the receiver buffer requirement versus packet-loss probabilities ranging from 0.001 to 0.1. The result shows that FEC has the lowest receiver buffer requirement under the operating conditions as set forth in Table 1. ARQ generally requires more receiver buffers than FEC and the hybrid algorithms except for loss probabilities below 0.015 and over 0.07. As we choose the hybrid algorithms' operating parameters to minimize traffic overhead, the receiver buffer requirement fluctuates between the ARQ curve and the FEC curve for different loss probabilities. If it is desirable to incorporate the cost of receiver buffers, we can modify the optimization procedure in choosing R so that an integrated objective function taking into both traffic overhead and buffer cost is optimized, rather than based solely on $H_{hybrid}(R)$.

F. Server Buffer Requirement

The server buffer requirement plots (not shown) are similar to the receiver buffer requirement plots. This is because due to retransmissions, longer buffer-holding time at the receiver also implies longer buffer-holding time at the server. With the system parameters in Table 1, the server buffer requirements for all cases (i.e. ARQ, FEC, Active and Passive Hybrid ARQ/FEC) remain below 200 KB per stream for a wide range of parameters

(N=1…1000. $p_i$=0.0…0.1). Therefore a 100-stream server would need no more than 20MB of memory for error-control. Given the rapid price drops and capacity gains in memory chips, we can conclude that server buffer requirement will not be a significant factor in selecting error-control algorithms.

G. Heterogeneous Packet-Loss Probabilities

So far, the results are computed with all receivers having the same packet-loss probability. In practice, different receivers will likely experience different packet loss probabilities because of different network conditions, different processing power, etc. To evaluate the performance of the proposed hybrid algorithms under heterogeneous receiver conditions, we set the receivers' packet-loss probabilities according to a uniform distribution over a range $P_L$ and $P_H$. Specifically, the packet-loss probability for receiver I is given by

$$P_i = P_L + (P_H - P_L)\frac{i}{N-1} \tag{66}$$

where $P_L$ and $P_H$ is defined by the average packet-loss probability $P_M$ and the range factor $\gamma$.

$$P_L = P_M(1-\gamma) \tag{67}$$

$$P_H = P_M(1+\gamma) \tag{68}$$

Setting $P_M$=0.25, we plot the traffic overhead for $0 \leq \gamma \leq 1$ in Fig. 13. As expected, ARQ is relatively insensitive to receiver heterogeneity. By contrast, FEC's performance deteriorates for increasing receiver heterogeneity because a fixed amount of redundancy is used for all receivers. The proposed hybrid algorithms have lower traffic overhead than both ARQ and FEC for the entire spectrum of $0 \leq \gamma \leq 1$. Again, the active recovery mode performs better than the passive recovery mode due to the better utilization of redundant packets.